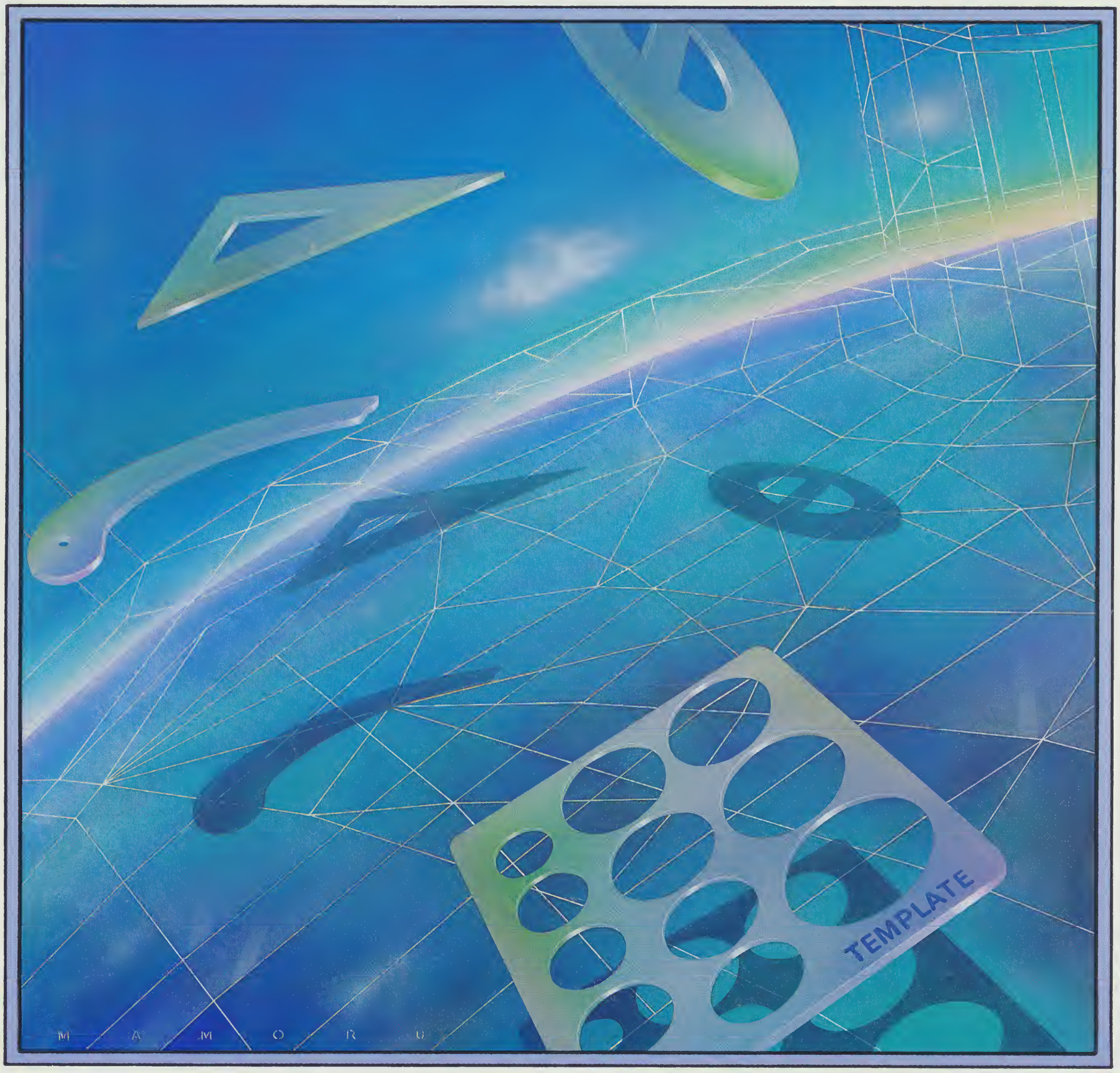
Computer Graphics World

APRIL 1981

VOLUME 4, NUMBER 4



SPECIAL SOFTWARE ISSUE

$3.00


M A M O R U

Device-Independent Graphics Software Comes of Age

By Dr. John M. Moreland
and Dwight C. Olson
Megatek Corporation

The computer graphics industry is coming of age and this is causing a tremendous surge of interest among potential users who have not previously been professional graphics programmers. To meet the application programming needs of these new users, Megatek has developed a high level general purpose graphics software support system called Template™.

Template was designed to provide both computer and device independence in a graphics application development package, with support for dynamic and static applications in both two- and three- dimensional environments. Its features include line drawing and text generation in both 2D and 3D using high level FORTRAN functions, user-definable viewing environments, structured archiving of graphics objects, general axis generation, color definition and selection, display of 21 character fonts, and a virtual surface capability.

Usable on any 24-bit or larger computer, Template supports virtually any commercially available graphics terminal device—from vector refresh, storage tube and color raster displays to flat-bed or drum pen plotters, COM recorders, alphanumeric line printers and interactive alphanumeric CRT terminals. Template is a comprehensive and modular collection of ANSI compatible FORTRAN subroutines which incorporates the CORE system concepts developed by the Graphics Standards Planning Committee of ACM SIGGRAPH and being used by the X3H3 ANSI Committee on Graphics Standards.

Advances in Computer Graphics Technology

The evolution of computer graphics technology is following a pattern similar to that followed by computer technology generally. Early developments were in the area of hardware innovation. First came the development of batch-processing oriented plotting devices. These were slow, and did not allow interactive input from the operator. Then in the 1960's, the advent of the storage tube display made computer graphics displays available to a broad range of users. The displays were still slow, by today's refresh oriented standards, and they were essentially non-interactive, but speeds were increased significantly over pen plotters and other mechanical plotting devices.

But in scientific and engineering applications, the real advances were yet to come. For example, the most common graphics display used in CAD/CAM systems was the storage tube, primarily because of its low cost. But a tradeoff had to be made for that low cost, because storage tube terminals did not offer multiple colors, were slow in drawing screen images, and offered little dynamic interaction between operator and terminal for on-line design and analysis tasks. Most importantly, the storage tube did not allow "selective erase." The nature of the storage tube re-

quires that once a drawing has been made on the screen, the entire picture must be re-drawn if any element of the picture is to be changed. This means that while an engineer can draw high-resolution designs, the process is slowed tremendously by the inability to interact dynamically with the screen image and to change it easily for design analysis. This has led to a "hit repaint and drink a cup of coffee" syndrome among design engineers.

Historically this has required a significant reprogramming job. The software required to drive a refresh display might have little resemblance to that needed to drive a penplotter, since the capabilities and protocols for these divices differ greatly. The result is that the same software development job might have to be done two, three or more times, depending on the number of kinds of graphics display devices that need to be supported. The situation is complicated further when the various display and plotting devices are attached to different host computers. The problems of moving a graphics application and associated data from one computer to another can become insurmountable.

"...Template allows a graphics application programmer to write his program just once, without worrying about which graphics device will be used when the program is executed."

The problems of low productivity were alleviated with the introduction of high-quality stroke refresh systems, allowing selective erase and local handling of interactive peripherals. And when these capabilities were augmented with the capabilities of real-time dynamic raster color, the engineer had the power to increase his productivity as never before.

Graphics of the Future

It is clear to those who have been involved in the evolution of computer graphics technology that the long-term costs to the user are predominantly those of the software his application requires, rather than the hardware. The costs of producing software have risen dramatically and the associated costs of maintaining and updating existing software have kept pace. During the life of a typical applications project, the cost of the hardware required to do the job will become insignificant when compared to the cost of software development and maintenance.

The situation is especially complicated for programmers working in the graphics display area. It is frequently necessary to move an application from one display device to another. For example, data may initially be displayed or an object modelled on an interactive refresh display. Later, it may be necessary to plot the same data or create a hardcopy of the modelled object on a penplotter or other display device.

It became clear that this was the single most important problem to attack. Template is the result of a major effort to solve that problem. The result of nearly 100 man-years of research, design, and development effort, Template allows a graphics application programmer to write his program just once... without worrying about which graphics device will be used when the program is executed. In a way completely transparent to the user, Template supports virtually every sort of display and plotting device. The device need not be specified until runtime.

In short, it provides a means of significantly reducing the software development costs associated with a wide range of engineering and scientific applications.

Design Philosophy

The intention of the designers was to provide for user convenience and ease of use. Features have been designed into Template to minimize restrictive rules, keep argument lists short, and employ only floating point numbers and Hollerith character strings. System-detected errors are also non-fatal.

The Template system is totally three dimensional. All 2D commands are executed in 3D via a user-specified "Z-value" which is set parametrically and which indicates the current X-Y plane. Two-dimensional coordinates thus refer simply to positions on that plane. Since the current X-Y plane can itself be positioned anywhere in 3-space, planar objects created by 2D commands can be produced in 3-space. However, if no 3D functions are executed because of the default settings, Template will execute as a 2D system—an approach adapted for use in the ACM SIGGRAPH CORE System.

The heart of the system is a central communication common block called the Graphics Status Area (GSA). Besides containing descriptive information about the display device, the computer, and the internal control information, the GSA also contains the modes and parameters which dictate the action of Template. Two basic routines are provided for setting the values of discrete modes and continuously variable parameters.

Template subroutines may be considered to fall into three categories: device and computer-independent, device-dependent, and computer-dependent. The device and computer-independent routines are the most numerous and implement most of the user-callable functions. To actually produce output, the device and computer-independent routines call the device-dependent routines to perform functions available in specific display devices. Template thus can easily access and utilize these functions when supported by the display-device. Both the device-independent and device-dependent routines utilize the computer-independent routines for functions not directly supported by ANSI Fortran or in which the implementation varies from computer to computer. These include byte packing and un-packing, random and sequential file I/O, and character conversion. The computer-dependent functions are device-dependent. Selection of devices occurs at program load time and consists of specifying the device-dependent li-

brary associated with the device to be used.

Megatek has been a strong participant in the current efforts to standardize computer graphics software languages. This began with the ACM SIGGRAPH efforts by the Graphics Standard Planning Committee resulting in the CORE system. Currently, support is being given to work by ANSI and the International Standards Organization to implement standards nationally and internationally. It is only through these efforts that the increasing cost of software development can be managed and kept to a minimum. Template has been designed according to the latest standardization efforts of these groups.

These are some of the capabilities and features implemented in Template:

Coordinate Systems

Template is a coordinate system-based graphics system. All reference to positions either in the user's plotting space (virtual space) or the display surface plotting space (device space) are based on the coordinate systems of those spaces. In virtual space, the basic coordinate system is a 3D righthanded coordinate system known as the "world" coordinate system. The domain of the world coordinate system is the range of floating point numbers on the computer being used.

Two types of user coordinate systems are maintained. The "reference" coordinate system is a cumulative coordinate system and the "working" coordinate system is non-cumulative. Reference coordinate systems are convenient to interrelate one coordinate system with another. For example, in astronomy, one coordinate system may be used for a plane, while another coordinate system may be used for the solar system which contains it. Template maps the one to the other automatically. Working coordinate systems are useful for temporary use such as positioning windows when drawing a house.

The coordinate system associated with the display device is known as the "device" coordinate system. Its main function is to specify the location of the view port, although it is possible to plot in this coordinate system when desirable. The coordinate system is a lefthanded coordinate system in which the X-axis extends from left-to-right on the display surface, the Y-axis extends from bottom-to-top, and the Z-axis (if the device has one) recedes into the device surface from the viewer. The origin of the coordinate system is in the lower-left corner and it may be referenced in several different units of measurement.

"During the life of a typical applications project, the cost of hardware...will become insignificant when compared to the cost of software development and maintenance."

Viewing Environment

In computer graphics systems, it is not sufficient to create a picture. The picture must be related to a view which dictates what will be physically displayed on the display surface. This requires that the location of the viewer and the direction of the view be defined. With Template, this "viewing vector" is defined by specifying one point in the user's virtual space indicating the location of the viewer (the viewpoint) and another point in the user's virtual space at which the viewer is looking (the view site). A vector emanating from the view point towards and passing through the view site is the viewing vector. It is also necessary to specify which direction in the user's virtual space represents the "up" direction. In addition, a plane perpendicular to the viewing plane will be located along the viewing vector to create a rectangle or window on the projection plane for clipping the projected image.

For 2D applications, the viewing environment may be set up to simulate a 2D graphics system. This is accomplished by placing the view site and the projection plane at the origin. The view point is placed somewhere on the positive Z-axis and the user-settable Z-value is set to 0. Since the viewing vector is along the Z-axis, the positive Y-axis represents "up" and 2D commands will product output only on the projection plane. Output thus will appear to two-dimensional.

Drawing Commands

Once the coordinate system and viewing environment have been chosen and defined, the user may create pictures using the Template drawing commands. Flexibility is provided through the use of either "relative" or "absolute" coordinates as well as combinations of relative for some coordinate components and absolute for others. The user may also select the type of coordinates to be rectangular, polar, cylindrical, or spherical.

A central line drawing routine can process 40 different line options. Variations within some options can increase the different styles of lines almost indefinitely. Lines are generated from the current position (end of the last line) to the indicated coordinates which become the new current position. The line options are created from five basic line types: Null, Line (solid), Dashed, Ticked, and Alpha. Each of these can be associated with any of eight line terminator types and variations include specification of dash segments, tic intervals, alpha character selection for Alpha lines and character terminators, and marker selection.

Three curve routines exist for generating curvilinear lines, including both circular curves and conic sections. These routines use hardware arc generators, if possible, otherwise the curved lines are simulated using straight lines. Curved lines may be considered to be either single lines or as collections of straight lines where each segment will contain a line terminator.

All visible output, whether lines or text, can be affected by the current attribute settings for color, intensity, and line width. In each case, the attributes are only effective if the hardware can support the particular attribute.

Text display

The Template text display facility is both powerful and convenient. Five

formats are supported: integer, real numbers, X-Y coordinates, X-Y-Z coordinates, and text strings. In each case, Template will perform all formatting necessary to display the desired text. For the first four types, the numbers provided are converted to character strings. In the last type, the characters in the provided string are displayed until a user-assignable text terminator character is encountered.

All textural output may be left, right, or center justified both longitudinally and transversely on the justification position. Template produces all output on the current X-Y plane indicated by the Z value. By suitable user coordinate system specifications, output may be produced anywhere in the user's virtual space.

Three types of characters can be produced. Hardware characters are generated by the device's hardware character generator, if one exists. Simulated hardware characters act exactly as hardware characters except they are produced by vector strokes. Both types are positioned to the justification position but will be produced on the display surface plane. Software characters are produced by drawing vectors in the current user coordinate system on the X-Y plane designated by the Z-component of the justification position.

Options are available for specifying individual character orientation, character italicization, character size and spacing, and character string angle from the X-axis. Support for these options in hardware character generators can be utilized but, with the exception of character size, is not simulated. Computer-independent upper/lower case shifting is provided for computer systems which do not have lower case characters. Subscripting and superscripting are available including in-line shifting for text strings. There are also text processing routines to reposition the current position to its previous value and to use the current position as the justification position.

Graphics Structures

A feature often desired by the users of graphics systems is the ability to collect groups of graphics commands into an entity which can later on, or in subsequent runs, be invoked with changes in position, scaling, rotation and mode setting. Moreover, such entities should be archivable in libraries. The Template Graphics Structure facility provides this capability. Structures are basic commands which have been saved as elements containing a command identification and the original input arguments. When the structure is invoked, these commands are re-executed in the current mode environment. The invocation contains a coordinate system definition to position the structure in the current picture space.

While being defined or invoked the structures are maintained in a random work file. If the user desires to save a structure library, a utility routine is available to reformat the structure as card images, which can be reloaded into the work file during subsequent runs. Utilities are also available to delete or rename structures and to merge structure libraries. Since the files are card images, they are computer-system independent and are a means of transmitting graphical images between Template installations.

Picture Organization

The concept of logically dividing the displayable output into segments can facilitate the implementation of some applications and is mandatory for using the selective erasure capability of refresh display devices. The Template segmentation facility allows users to divide the displayable output into named and numbered segments. While segmentation is a feature primarily provided to support selective erase and picking on refresh displays, it is also useful for producing backgrounds for movies or slides.

Associated with each segment are segment attributes which control visibility, highlighting and pick detectability. Each of these may be turned on or off individually for each segment as long as the segment exists. Segments may also have a type which indicates the kind of image transformations which may be applied at the display device, assuming the device in use has local transformation capability. Automatic double buffering is incorporated within the segmentation facility to allow existing segments to be displayed while they are being redefined. When a segment is no longer needed, it may be deleted.

> "A central line drawing routine can process 40 different line options. Variations within some options can increase the different styles of lines almost indefinitely."

Interaction Routines

One of the most significant difficulties associated with designing a device-independent computer graphics support system involves the handling of interactive graphics peripherals. There exists a wide variety of such devices, including data tablets and digitizers, joysticks, light pens and valuator devices. Many graphics applications make use of operator interaction through such devices in an essential way. In fact, the largest uses of computer graphics in the future will be in those applications involving interactive graphics input during program execution. As a result, any software support system which does not support such devices will not satisfy the needs of most users.

The problem is that different graphics display systems provide different types of interactive graphics peripherals, and some provide none at all. So how can these devices be supported in a device-independent way—in a way which doesn't require reprogramming to support different devices?

Template solves this problem by supporting a large variety of *logical* input devices, including pick, locator, keyboard, digitizer, valuator, and button. These logical devices

are then associated at program load time with the most appropriate available physical input devices. So input from a locator might come from a joystick if that physical device is available. But if at another time a tablet is available, it can be used as the locator device, and no reprogramming is required. This means the programmer need not concern himself with which devices are actually available, only with the logical functions that are to be performed.

The axis generation facility is used to automatically plot arrays of data and includes the ability to hold constant, or automatically increment, the values for any coordinate component if individual values for each point are not provided. If several curves are to be produced, any set of points may be automatically repeated if desired. Curve fitting and data averaging utility routines are provided which allow linear, least squares, spline and time series fitting, and many average adjustments.

"...the graphics programmer end user can focus on a small and powerful set of logical graphic utilities and let Template handle mapping these to the available physical devices."

Template's interaction routines support synchronous communication with the terminal operator. Input requests are initiated by calling one of several input functions which prompt the user for input, enable the device, wait for the input to arrive, transform the input to user form, and return it to the calling program. Input requests from the calling program selectively prompt the operator, echo input, transform the input as required, and then return it to the calling program. Segment picking and selection from a Template generated menu provides for additional program control.

Axis Generators and Plotting

A family of higher level functions has been provided in Template to facilitate the creation of axis systems and to use these axis systems to create plots, bar charts, grouped bar charts, histograms, pie charts, scatter diagrams, and time series plots. Each of these uses a central axis system generation facility which produces general 2D and 3D axes in 3-space or creates a view-adjusted 2D axis system where the viewing environment is altered so that the desired axis pair will appear on the projection plane. Label options include the independent selection of numeric labels for the tics on each axis and alphanumeric titles along each axis. Position and format of the labels may be controlled as well as their orientation with respect to the axes.

Other Features

One of the most significant features of Template is the virtual display surface facility. Frequently, pictures which are ultimately destined for a specific medium (e.g., 35 mm color slides) are designed on a more interactive display device. Template provides a mnemonic format specification for the production medium. When a format is selected, the display surface of the currently selected display device is configured to produce this format. If it cannot produce the format directly, a subsection of the physical display surface is used to simulate the requested media by providing a formatted display surface occupying the largest rectangle which provides the correct aspect ratio. The display surface dimensions are also set so that the actual dimensions of the requested media are supported. Examples of formats provided are 35mm color slides, 16mm movies, A through E size engineering drawings, and 8½ by 11 inch pages. It is also possible to specify the exact dimensions of the formatted display surface. The flexibility provided by this feature can increase productivity of quality slide-production and related applications significantly.

Another feature is support for user-definable color look-up tables. The normal color selection techniques provide an index for the color table. For some devices the color table consists of fixed color specification (in the case of monochrome systems, the color table contains only one entry). Template allows user specification of color table entries for those display devices which support loadable color tables, such as raster CRT systems and COM recorders.

Finally, Template supports a pseudo-display device facility. It produces output on the currently selected display device. The user chooses this device at the time he loads his program. The pseudo-display device is a file which may be substituted for actual display devices at this time. Output produced by Template is stored in a device-independent fashion on the file. This output may then be reproduced on a real display device in a separate program to read the pseudo-device file and copy it to the now currently selected device. This allows display data to be easily moved from computer to computer and from display device to display device without forcing re-execution of the program which produced the data.

Conclusion

It has not been possible here to detail all the capabilities of Template. However, enough has been provided to indicate the broad range of features provided. These features provide the graphics programmer and user with the tools necessary to solve virtually any application problem, whether it be plots of data, contouring, 2D and 3D modeling, or any other need. Providing these tools in a computer and device-independent way, means software development and maintenance costs are reduced even further. Rather than concern himself with an unending series of graphics display, plotting and input devices, each with its own distinct software protocol, the graphics programmer end user can focus on a small and powerful set of logical graphic utilities and let Template handle mapping these to the available physical devices.

It is only through software tools such as Template that the engineering and scientific uses of computer graphics can fulfill their ultimate promise. ■

MEGATEK CORPORATION

World Headquarters
3931 Sorrento Valley Blvd.
San Diego, CA 92121
714/455-5590
Telex: 910-337-1270
Eastern Regional Sales Office
Boston, MA
Southern Regional Sales Office
Atlanta, GA
Western Regional Sales Office
San Diego, CA